Human action recognition using combined contour-based and silhouette-based features and employing KNN or SVM classifier

[1]Salim Al-Ali, Mariofanna Milanova, Agata Manolova, and Victoria Fox

***Abstract*—**This paper presents a new algorithm for human action recognition in videos. This algorithm is based on a combination of two different feature types extracted from Aligned Motion Images (AMIs). The AMI is a method for capturing the motion of all frames in a human action video in one image. The first feature is a contour-based type and is employed to grasp boundary details of the AMI. It relies on the 1st and 2nd discrete time differential of the chord-distance signature feature, so it is called Derivatives of Chord-Distance Signature (DCDS). The second feature is a silhouette-based type that is used to capture regional appearance details. It catches most of the visual components for the AMI using a Histogram of Oriented Gradients (HOG) feature. Combining both features creates a complementary feature vector that makes it possible to obtain an optimal correct recognition rate of 100%. For the classification, the algorithm is utilized two different classifiers: K-Nearest-Neighbor (KNN) and Support Vector Machine (SVM). The KNN is based on the 1st norm distance and achieves slightly better results than this obtained by SVM. The performance of the algorithm is tested through six experiments. Three experiments for the KNN classifier and others for the SVM. For each classifier, three experiments conducted to determine the effectiveness of each feature separately and when combined. The experimental results demonstrate the potential power of this algorithm and its promising success in human action recognition in videos.

***Keywords*—**Contour-based, human action recognition, video recognition, silhouette-based.

I. Introduction

HUMAN action or human activity recognition research has captured more attention due to the variety of useful applications presents in different computer fields, such as human-computer interaction, surveillance environments, robotic machines, healthcare systems, multimedia retrieval, and entertainment environments [1, 2, 3].

According to Aggarwal and Ryoo [1], human action recognition is classified into two main approaches: single-layered and hierarchical. The single-layered approach depends on a sequence of images to describe human actions while the hierarchical approach depends on simple actions to describe the human actions. On the first hand, the single-layered approach is used to represent the human action and is categorized based on a model as: space-time and sequential methods. On the other hand, the hierarchical approach is categorized based on the methodology that is used to recognize the actions as: statistical, syntactic, and description methods. The space-time approaches are classified based on the type of features, which are obtained from the space (spatial) and time (temporal), into space-time volume features, trajectories, and space-time features. Our research is based on the single-layered approach and space-time volume features.

Gorelick et al. [3] employed contours of silhouettes extracted from the Weizmann dataset [2] as space-time volume features for human action recognition. The authors solved Poisson's equation depending on the contour coordinates and found coefficients of this equation by using a multigrid solution. Variant of properties are extracted based on these coefficients, such as local space-time saliency, action dynamics, shape structure, and orientation. Next, these shape properties (Poisson features) are used as a sequence of features for a sliding overlapped window of frames in each video. Finally, for the classification, the variant median Hausdroff is used to find the distance between any two sequences.

Whytock et al. [4] achieved good recognition results in terms of accuracy. The authors combined the Gait-Energy Image (GEI) and Histograms of Oriented Gradients (HOG) descriptors for the action recognition. For HOG, they used more than four different gradient filters--Lele, Fourier-Pade-Galerkin, Bicklley, and Scharr schemes--in addition to the standard central difference scheme and Sobel kernel. They used SVM classification based on the leave-one-sequence-out cross validation technique and tested their algorithm over the Weizmann dataset. Accuracy is achieved by decomposing actions into static and dynamic classes. For the Weizmann dataset, the authors defined static actions as one-hand wave, two-hand wave, bend, jump in place, and jumping jack, while dynamic actions are run, walk, skip, jump, and gallop sideways. They compared the performance of one verse all (OVA) and one verse one (OVO). Also, they used the SVM with five kernels: Linear, Quadratic, Polynomial, Gaussian Radial Basis Function, and Multilayer Perceptron. In this paper, however that all static and dynamic actions are used as

[1] Salim-Al-Ali is with the Computer Science Department, University of Arkansas at Little Rock, AR 72204 USA, (e-mail: sgsaeed@ualr.edu).

Mariofanna Milanova is with the Computer Science Department, University of Arkansas at Little Rock, AR72204 USA (e-mail: mgmilanova@ualr.edu).

Agata Manolova is with the Faculty of Telecominication, Technical University of Sofia, Bulgaria (e-mail: amanolova@tu-sofia.bg).

Victoria Fox is with the University of Arkansas at Monticello, AR 71655 USA (e-mail: fox@uamont.edu).

one set (without any decomposing), our algorithm achieved an optimal result of 100%.

Sadek et al. [5] combined chord-length or chord-distance with the center of gravity as a feature and employed SVM as a classifier for human activity recognition. The chord-distance was used to describe the shape. Based on the authors' report, these shape features are invariant to translation, rotation, and scaling. Nevertheless, these descriptors are neither sufficient nor compact enough to be used alone, and, therefore, they are constantly added on a reference point called the center of gravity global feature. Thus, the authors fused the shape descriptors with the global features for motion to form the final SVM model with the Gaussian kernel. In this research work, only two features are employed without any global feature.

This paper is an extended version of our conference paper [6]. In this paper we extend the experiments to determine the effectiveness of each feature separately and when combined. Briefly, we test our algorithm in six experiments. These experiments are based on two features (contour and silhouette) and two classifiers (KNN and SVM). The first experiment is based on the DCDS feature and the KNN classifier. The second is based on the HOG feature and the KNN. The third is based on combined both features and the same classifier. The fourth experiment is based on the DCDS feature and the SVM classifier. The fifth is based on the HOG feature and the SVM. The last experiment is based on the combination of both features and the SVM classifier. The combination of these two features formed a complementary feature vector. The SVM is performed slightly better results in the experiments of using each feature separately, but the KNN achieved a slightly better result than the SVM in using of the combined features.

The rest of this paper is organized as follows. Section 2 reviews related literature. The details for our algorithm are provided in Section 3. The experimental results for six different experiments are discussed in detail in Section 4. Finally, the conclusions are presented in Section 5.

II. Related Literature Reviews

This section provides related topics that were used to create the presented algorithm: Aligned Motion Image (AMI), Fourier Descriptors (FDs), Histogram of Oriented Gradients (HOG) feature, and Derivatives of Chord-Distance Signature (DCDS) feature. Moreover, the classifiers K-Nearest Neighbor (KNN) and Support Vector Machine (SVM) are explained in detail.

A. *Gait Energy Image*

The GEI [7] is the average accumulation for motion image that captures motion with different intensity values for each pixel in the image. Therefore, the GEI grasps two features in each pixel of frames in the video. First, the spatial holds the location of the pixel. Second, the motion captures the average use of this pixel, which is represented by intensity value. The GEI is calculated by using a function such as the following:

$$G\tau(x,y,t) = \frac{1}{\tau}\sum_{t=1}^{\tau} F\,(x,y,t) \qquad (1)$$

where x, y denotes the position of a pixel in a frame; t is the time, which represents the frame number in a video clip; τ is the number of frames in a video clip and represents duration of the video clip, and F is the t frame of binary data.

In our research, the AMI was formed by merging a sequence of aligned frames in a video-- one over another. It was inspired by the GEI, but the primary difference is that, in this work, all frames in the AMI were aligned. This image captures motion details through all frames in the video. Also, it captures the spatial details of each frame among frames in the video. Therefore, the AMI contains the most important details for each action. [The term "aligned" is used to indicate that silhouettes which were extracted from all frames are aligned at the center of the frame (centroid).] Figure 1 depicts examples of AMIs for all actions of the Weizmann dataset. The silhouettes are extracted by applying background subtraction. The backgrounds are available in the Weizmann dataset, if the backgrounds are not available, other methods can be applied [8, 9].

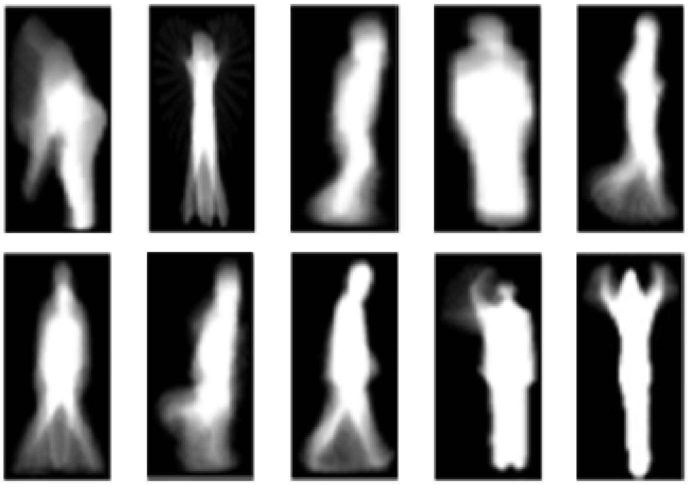


Fig. 1 Examples of AMIs for human actions in the Weizmann dataset: (top row) bending, jumping jack, jumping, jumping in place, and running, respectively; (bottom row) side jumping, skip jumping, walking, one hand waving, and two hands waving, respectively

B. *Fourier Descriptors (FDs)*

The FDs are based on discrete Fourier transform (DFT), which is a mathematical operation for converting a function of time domain into frequency domain. In short, FDs are used to describe the contour (boundary) of any closed shapes in 2D space depending on discrete Fourier transform methods. The importance of FDs is due to their representation of any 2D closed shapes independent of location (translation), scaling, rotation and starting point (FDs properties) [10, 11]. Thus, the motivation for using FDs is related to these properties. Moreover, it useful to unify the number of points that represent each shape boundary of the AMI. This benefit is achieved by the approximation in the reconstruction of the FDs.

In our research, the function of FDs and the inverse function of FDs were used based on the methods of Gonzalez et al. [12]. The FDs are represented by imaginary numbers:

$$z(t) = x(t) + iy(t) \tag{2}$$

where z is a complex number function, [x(t), y(t)] are Cartesian boundary points of a contour in 2D space, t is an integer such that t∈[1..N], N is a number of points on boundary, and symbol (i) refers to the imaginary part of the complex number. The FDs function F is calculated as follows:

$$F(k) = \frac{1}{N}\sum_{t=1}^{N} z(t)e^{-j2\pi tk/N} \tag{3}$$

where k is an integer such that: 1≤k≤N, e is the exponential function. In order to reconstruct function z(t), the inverse of DCT, which is z^' (t), based on F(k) is computed:

$$z'(t) = \sum_{k=1}^{N} F(k)e^{j2\pi tk/N} \tag{4}$$

However, the approximation of z can be reconstructed by using the function z^' (t) with fewer Fourier coefficients such that1≤k≤p and p<N. This approximation is important to unify the number of points for all contours of the AMIs.

The FDs are employed to define coordinates of the contours. They capture the most important details for the boundary that surround the extracted object for the AMI. Moreover, another useful benefit is their ability to unify the number of contour coordinate points in all AMIs, since these points are different in terms of number of coordinates among all AMIs.

The process of reconstructing 32 FDs is shown in Figure 2. It is obvious that the image reconstructed by FDs is similar to the original image except that it captures the most important and ignores the less important details.

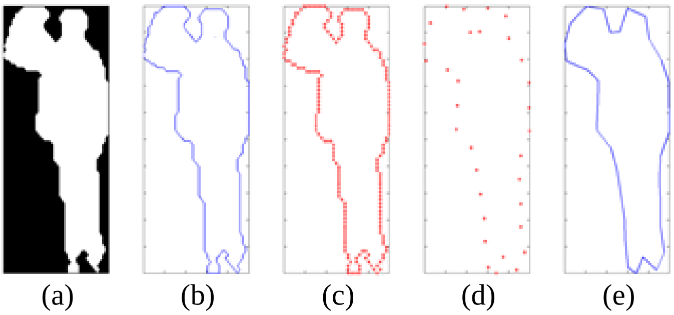


Fig. 2 Calculating FDs for one hand action from Weizmann dataset, (a) silhouette of the AMI, (b) contour of the silhouette, (c) all Cartesian coordinate points of the contour, (d) 32 points reconstructed from FDs, (e) plotting of the 32 points.

C. Chord-Distance Signature (CDS)

The CDS [13] is a function obtained from the distance (magnitude length) between two points on the contour (shape boundary) of the silhouette. These distances are used as compact and robust features in recognition systems [5]. The distance is controlled by an integer (w), which is a jump displacement step, in terms of the number of points, that separates two points on the reconstructed contour. The calculation of CDS function is as follows:

$$CDS(t) = \sqrt{(\nabla x)^2 + (\nabla y)^2} \tag{5}$$

where CDS is the chord function, t is an index integer such as t∈[1 ..N], and N is the number of coordinate points on the reconstructed contour. ∇x=x(t)– x(t + w) is the difference in the x-coordinate values; ∇y=y(t) – y(t+w) is the difference in the y-coordinate values; (x, y) are the Cartesian coordinates in 2D space for the contour, and w is the jump displacement between all pairs of contour points. Figure 3 shows the process of calculating a CDS example for bending action in the Weizmann dataset.

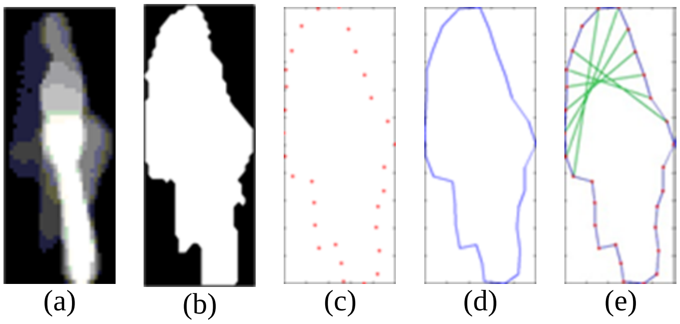


Fig. 3 Calculating CDS example for bending action from Weizmann dataset, (a) the AMI for bending, (b) silhouette of the AMI, (c) 30 reconstructed FD points for contour of the silhouette, (d) plotting of these points, (e) some CDS examples (the green lines) with jump displacement points (*w*=8).

D. Derivatives of CDS (DCDS) feature

The motivation for using the derivatives is that the first and second derivatives capture the most important details for edges of contour shape [14]. After calculating all *CDS* distances between all pairs on the contour, the first and second derivatives for these distances are computed using discrete time differential in image processing. The approximation of the 1st derivative of *CDS*, which is CDS' , is shown as follows:

$$CDS'(t) = CDS(t+1) - CDS(t) \tag{6}$$

where *CDS* is the chord function, *t* is an integer index for *CDS*. Also, the approximation of the 2nd derivative of *CDS*, which is CDS'', is shown as follows:

$$CDS''(t) = CDS'(t+1) - CDS'(t) \tag{7}$$

where CDS' is the 1st derivative of *CDS* function, and *t* is an integer index for *CDS*.

Finally, both derivatives, CDS' and CDS'', are combined to form one feature vector called DCDS.

E. *Histogram of oriented gradient (HOG) feature*

The HOG is a robust feature descriptor for the shape of the object [15]. This feature is extracted from an image based on two parameters: the number of overlapped windows on the image (NxN) and the number of bins (B) for the gradient angles. First, the image is divided into NxN overlapped windows. The gradients of intensities for each window's pixels are computed by using a horizontal kernel [-1, 0, 1] and a vertical kernel [-1, 0, 1]-1. Next, the angles and magnitude are computed for each pixel in the window. Subsequently, the angles are divided into B groups based on the number of bins. The total sum of the magnitudes for each group is obtained. Next, this operation is performed for each overlapped window. Finally, after all windows are finished, NxNxB numbers are normalized. These numbers are the HOG feature for the image.

In our research, the HOG feature was computed using the AMI image for each video. The parameters were set to 3x3 overlapped windows (N=3) and 8 bins (B=8), which achieved the best result in our experiments in terms of accuracy.

F. *K-Nearest Neighbor (KNN) classifier*

The KNN is the simplest method used for classification, clustering, and regression [16, 17]. This classifier is used in machine learning, pattern recognition, and data mining. It obtains class membership for some testing feature descriptors based on its nearest neighbor from training feature descriptors. The testing is classified by a majority vote of its K nearest neighbors.

In the KNN, there are three parameters. The first parameter, K, is set to the number of voting members. The second, distance metric type, is set to Euclidean, cityblock (absolute difference), cosine metrics, etc. The third parameter, the rule for selecting an estimated class for the testing sample, is set to the nearest neighbor, random, etc. The KNN classifier is calculated using the distances between the testing video and each training sample, as provided by Equation 8:

$$d(x, m_j) = \arg\min_j \{d(x, m_j)\} \tag{8}$$

where d is a distance metric, x is a testing sample, m are training samples, j = [1, 2, …, N], and N is a number of training samples. The distance d is arguably the minimum distance (nearest neighbor) among distances between x and each training sample. The class membership for action with minimum distance is defined as a class membership for the testing sample. In all KNN experiments, the Leave-One-Video-Out (LOVO) cross validation technique is employed. Thus, all videos in the dataset are used for training except one video that is used for testing.

G. *Support Vector Machine (SVM) classifier*

The SVM is a binary classifier that separates some feature descriptors by an optimal hyperplane used as a decision function [18, 19, 20]. This hyperplane is represented as a separation; hence it is called a margin classifier, as shown in Figure 4.. The SVM can be used to perform a linear or non-linear classification based on the kernels used. Once the SVM is trained to recognize the features of training samples, the classifier can make decisions about some features in a testing sample regardless of the absence of these features in the testing sample. This classification is performed almost like a human behavior when making a decision.

In all SVM experiments, the Leave-One-Actor-Out (LOAO) is employed. It is a 9-fold cross validation technique; therefore, all videos are separated into the number of actors, which are nine sets. One actor (set) is used for testing and the others for training. In this experiment, the dataset is separated into 9 folds. Each fold represents one actor in the dataset videos. The classification is repeated 9 times. Each time, one fold (actor) is used for testing, and others are used to train the classifier. By the end of all times, all videos are used in testing and training modes, and the average of the correct recognition rate is computed as a result of the recognition for this experiment.

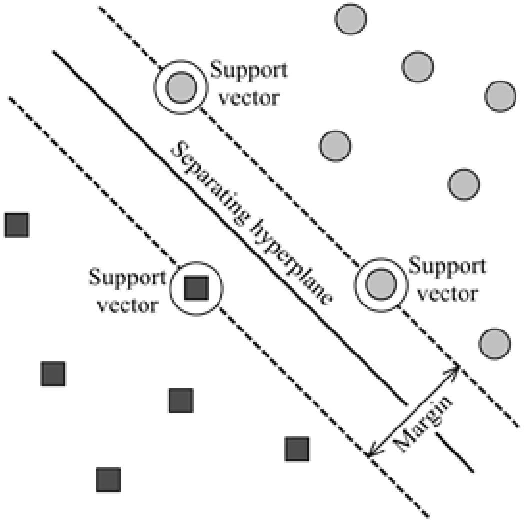


Fig. 4 Linear Classification of SVM classifier

III. The Proposed Algorithm

The proposed human action recognition system consists of two modes. First, the training mode is a program used to train the system about the human actions using the training video samples. Second, the testing mode is a program employed to test a video with unknown human action and identify (classify) what kind of human action is happening based on the training video samples.

A. *Training mode*

The training mode is always the initial stage in human action recognition. This mode consists of several processes: reading, computing AMI, computing contour feature (DCDS), computing silhouette feature (HOG), building feature vector, and saving the feature. These steps are repeated for a number of the training video samples. Figure 5 shows the main structure of the training mode.

The training mode is started by the reading process, which reads a video, frame by frame, from the training dataset. Note that the aligned masks of the extracted objects from the

Weizmann dataset [2] are used. Thus, there is no need for pre-processing steps to extract silhouette objects, such as background subtraction, thresholding, and aligning (centroid) objects. The alignment has an important benefit when building the AMI: it makes it possible to eliminate differences among videos in terms of the number of frames in each video sample. This is true especially for actions that have moving displacement, such as running, walking, jumping forward, etc. However, at the same time, the alignment has a deficiency, which is considered an additional pre-processing step for recognition.

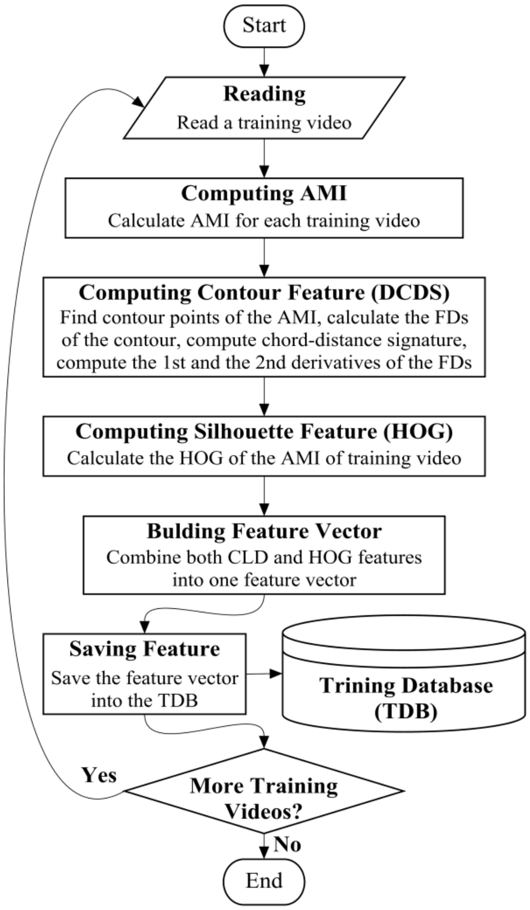

Start
Reading
Read a training video
Computing AMI
Calculate AMI for each training video
Computing Contour Feature (DCDS)
Find contour points of the AMI, calculate the FDs of the contour, compute chord-distance signature, compute the 1st and the 2nd derivatives of the FDs
Computing Silhouette Feature (HOG)
Calculate the HOG of the AMI of training video
Bulding Feature Vector
Combine both CLD and HOG features into one feature vector
Saving Feature
Save the feature vector into the TDB
Trining Database (TDB)
Yes
More Training Videos?
No
End

Fig. 5 Structure of training mode algorithm.

Computing AMI is the second process in this mode. The goal is to compute AMI from all frames in each training video sample. In order to calculate these images, the total summation frame of all aligned frames of silhouettes is computed by using Equation 1. Before dividing it into the number of frames in the video to form the AMI, a filtering is applied to reduce noise in this summation, where all pixels with 1 value in the summation frame are converted into zero (0) values. The influence of the filtering is high on the experimental recognition rate results because these pixels either appeared one time accidently during all the frames of a video or most likely just a noise added to one of the frames. Other pixels, which have values more than one (1), are most likely the result of natural human action motion in the video. The importance of the AMI can be summarized as follows: eliminating differences in videos in terms of the number of frames and forming images that will be helpful later on in forming discriminate features.

Subsequently, the process of computing DCDS feature starts with a few steps. First, the contour coordinate points are obtained for the AMI. Second, FDs are applied to these coordinate points to unify the number of these points since each contour boundary is different in terms of the number of points. In this work, the best result was recorded when the number of FDs was set to 30 points. Third, Equation 5 was used to calculate the CDS. Fourth, the 1st derivative of the CDS was computed using Equation 6, and the 2nd derivative was computed using Equation 7. After these four steps, the derivatives, which are 60 in number, were normalized and counted as the first group of recognition features.

Next, after computing the DCDS feature process, the HOG feature process was started by obtaining the bounding box, which is the smallest box that contains all pixels of non-zero values around the AMI object. Then, the image of the bounding box was extracted. After that, the HOG was calculated for extracting the bounding box. In this work, the HOG parameters (NxN and B) were set to 3x3 and 8, respectively. This means that 9 overlapped windows were used for HOG. Also, 8 bins were used, which means that the gradients between every angle with a 45 degree will be counted as one bin. By the end of this process, the results of HOG will be 3x3x8 numbers. These 72 shape descriptors are the second group of recognition features.

The final process, building one feature vector, is applied to combine both HOG and DCDS features to form one vector. Both groups are normalized in this process to have values between 0 and 1. After that, the feature vector is saved into a Training DataBase (TDB), which is a matrix where each row represents a feature vector of one training video sample and the number of rows represents the number of all samples. Each row contains (72+60), the numbers representing the DCDS and HOG features, respectively.

Finally, all of the steps in a training mode are repeated as many times as the number of available training video samples. After finishing all samples, the TDB will contain all feature vectors. Later, in the next testing mode, the TDB is used to train the recognition system about the human actions.

B. Testing mode

The testing mode is the second stage in human action recognition. The testing mode consists of the following processes: reading, computing AMI, computing contour feature (DCDS), computing silhouette feature (HOG), building feature vector classifying based on the TDB, and identifying the action that happened inside the testing video. The main structure of the testing mode is shown in Figure 6.

At the beginning, all of the processes of the training mode are repeated in the same manner to build the feature vector. These steps are the same in every detail between the two modes so that the comparison will be successful.

Next, after building the feature vector for the testing video, the classification process begins by using one of two different algorithms to classify this vector based on the vectors in the TDB, which have already been created in the training mode. These two algorithms are KNN and SVM; one of each is used in different experiments concerning human action recognition. For KNN classification, the 1st norm is used to calculate the distances between the feature vector of the testing video and each vector in the TDB. Also, for SVM classification, the LIBSVM 3.17 [20] is used with multi-class SVM type and linear kernel type, which provide better results than other SVM or kernel types.

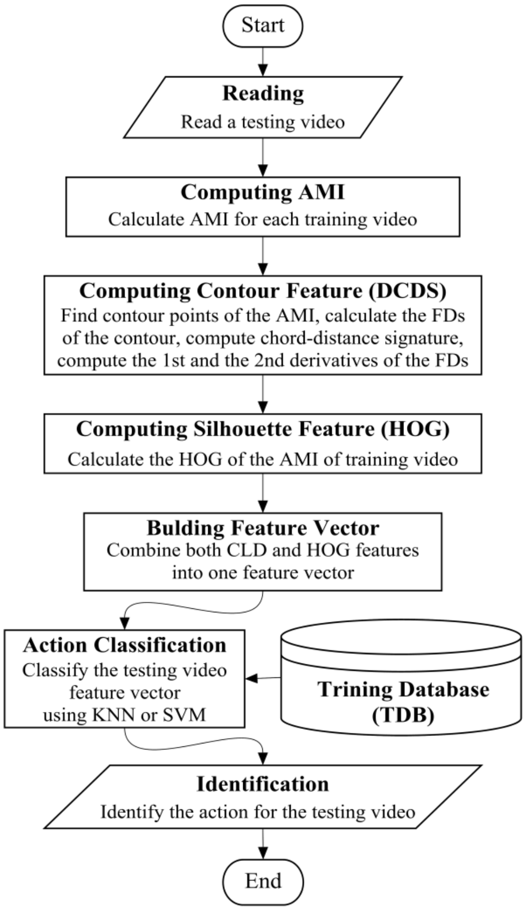

Start
Reading
Read a testing video
Computing AMI
Calculate AMI for each training video
Computing Contour Feature (DCDS)
Find contour points of the AMI, calculate the FDs of the contour, compute chord-distance signature, compute the 1st and the 2nd derivatives of the FDs
Computing Silhouette Feature (HOG)
Calculate the HOG of the AMI of training video
Bulding Feature Vector
Combine both CLD and HOG features into one feature vector
Action Classification
Classify the testing video feature vector using KNN or SVM
Trining Database (TDB)
Identification
Identify the action for the testing video
End

Fig. 6 Structure of testing mode algorithm.

Subsequently, the final identification process is used to identify the action that happened in the testing mode using the KNN algorithm or the SVM algorithm. In the KNN, the action in the TDB that its feature vector has minimum distance with the feature vector of the testing video is identified as the action for the testing video. In the SVM algorithm, the class is already identified by the LIBSVM.

IV. Experimental Results

In this section, the Weizmann dataset is described, and two different groups of experiments are presented. The first group experiment used the KNN as a classifier and the second employed the SVM. In both groups, three experiments were conducted. One used only DCDS as the feature for recognition. The second employed HOG feature only. In the third, a feature vector of both (DCDS and HOG) are combined and used. In the KNN experiments, the technique of Leave-One-Video-Out (LOVO) was applied where one video was used for testing, and all other videos were used for training at each testing time. For the SVM, the Leave-One-Actor-Out (LOAO) technique was applied in rounds. In each round, all videos of one actor were used for testing, and the others were used for training. These rounds were repeated for all actors, and the average was calculated as a final result for recognition.

A. Weizmann Dataset [2]

In this research, the Weizmann dataset was employed to test the algorithms. This dataset was created by Gorelick et al. It consists of 93 low-resolution (180x144) videos recorded at a speed of 50 fps. The dataset contains videos of 9 different people with each person performing 10 natural actions: bending, jumping jack jumping, jumping, jumping-in-place, running, side jumping (gallop sideways), skip jumping, walking, one hand waving, and two hands waving. One of these actors performed three (3) actions (running, walking, and skip jumping) twice. One action is performed from left to right, and the other from right to left. All videos were recorded by a static camera; hence, the background in the videos is static, which is very suitable for background subtraction processing. Note that, the number of frames in each video of the dataset is different. Figure 7 shows one frame example for each action in the dataset.

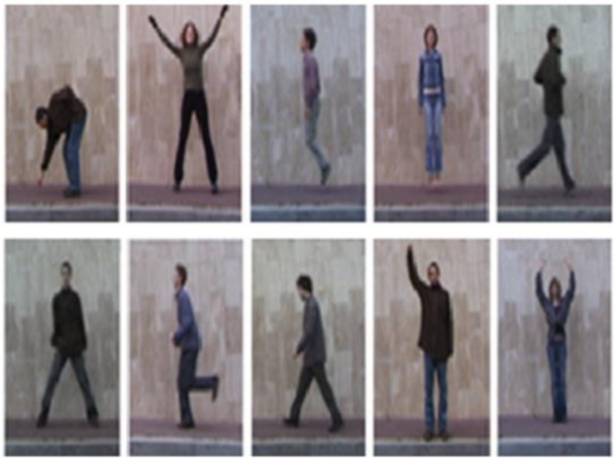


Fig. 7 Weizmann dataset frame examples for a different person performing a different human action: (top row) bending, jumping jack, jumping, jumping in place, and running; (bottom row) side jumping, skip jumping, walking, one hand waving, and two hands waving.

B. K-Nearest Neighbor (KNN) Experiments

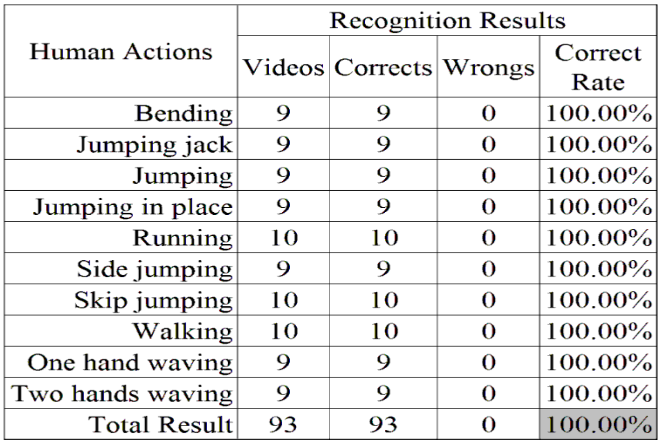
The KNN was used as a classifier in all of the experiments based on the LOVO technique. Two types of features were employed: the first is derived from contour-based type; the second is silhouette-based. Using the KNN classifier, three experiments were conducted. In the first experiment, only the contour-based feature was applied. The silhouette-based feature was utilized in the second experiment. In the third, both features were combined and used as one feature vector.

1. Both DCDS and HOG Features Experiment

In the first experiment, the KNN was used as a classifier, and both kinds of features (DCDS and HOG) were combined and employed as one feature for human action recognition in the Weizmann dataset. The setup parameters for the DCDS were as follows: the number of FDs points was set to 30, and the jump displacement among these points was set to 8 or 22 points. The setup parameters for the HOG were as follows: the number of the overlapping windows (NxN) was set to 3x3, and the number of bins was set to 8. Also, the 1st normal distance for the KNN classifier was utilized.

An optimal recognition rate accuracy of 100% was achieved, as shown in Table 1. It is obvious from these results that the combination of both DCDS and HOG led to an effective result in terms of the correct recognition rate.

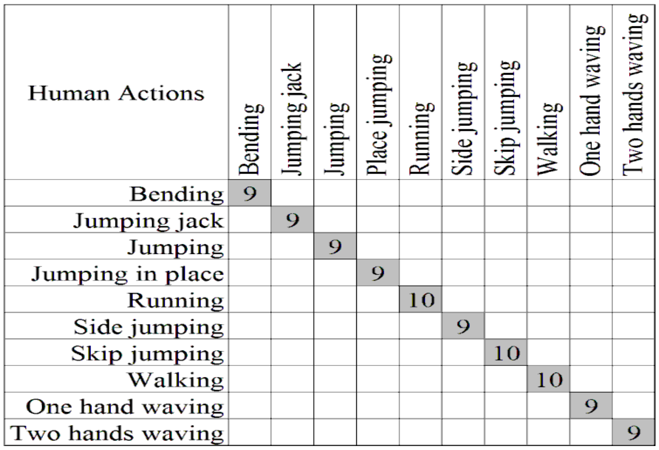
TABLE 1. Recognition results of combined DCDS and HOG features using KNN classifier



Human Actions	Recognition Results			
	Videos	Corrects	Wrongs	Correct Rate
Bending	9	9	0	100.00%
Jumping jack	9	9	0	100.00%
Jumping	9	9	0	100.00%
Jumping in place	9	9	0	100.00%
Running	10	10	0	100.00%
Side jumping	9	9	0	100.00%
Skip jumping	10	10	0	100.00%
Walking	10	10	0	100.00%
One hand waving	9	9	0	100.00%
Two hands waving	9	9	0	100.00%
Total Result	93	93	0	100.00%

Moreover, the confusion matrix listed in Table 2 proves that this result is optimal because there is no confusion among all of the actions in the Weizmann dataset.

TABLE 2. Confusion matrix results of combined DCDS and HOG features using KNN classifier



Human Actions	Bending	Jumping jack	Jumping	Place jumping	Running	Side jumping	Skip jumping	Walking	One hand waving	Two hands waving
Bending	9									
Jumping jack		9								
Jumping			9							
Jumping in place				9						
Running					10					
Side jumping						9				
Skip jumping							10			
Walking								10		
One hand waving									9	
Two hands waving										9

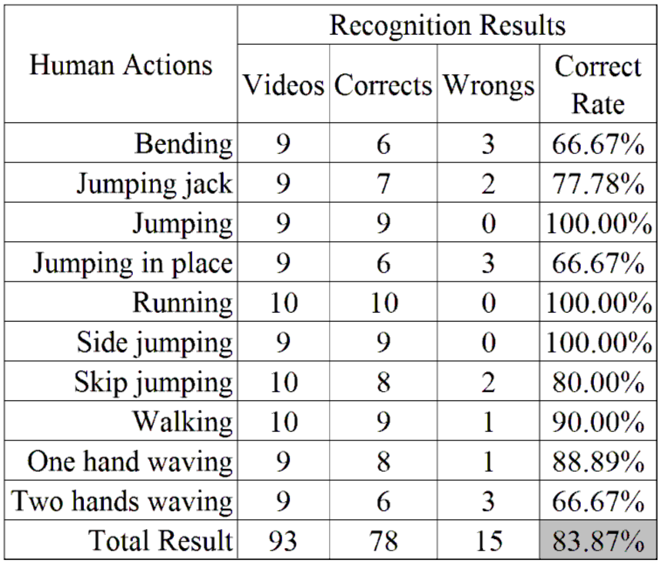
2. Only DCDS Feature Experiment

In order to determine the effectiveness of the DCDS feature separately (without HOG), this experiment was performed using only the DCDS feature.

This feature is a contour-based type obtained from the 1st and 2nd derivatives of the CDS feature. The latter is computed using the FDs for the contour of the AMI. The setup parameters for this DCDS experiment were as follows: the number of FDs points was set to 30, and the jump displacement among these points was set to 8 or 22 points. For classification, the KNN was employed based on the 1st norm distance. Note that, in this experiment, the setup parameters used were the same as those employed in the first experiment in order to make a fair comparison among these KNN experiments.

In this experiment, a correct recognition rate of 83.87% was achieved, as shown in Table 3. This table shows the experimental results for each action in the Weizmann dataset, as well as the total result for all actions.

TABLE 3. Recognition results of DCDS feature using KNN classifier



Human Actions	Recognition Results			
	Videos	Corrects	Wrongs	Correct Rate
Bending	9	6	3	66.67%
Jumping jack	9	7	2	77.78%
Jumping	9	9	0	100.00%
Jumping in place	9	6	3	66.67%
Running	10	10	0	100.00%
Side jumping	9	9	0	100.00%
Skip jumping	10	8	2	80.00%
Walking	10	9	1	90.00%
One hand waving	9	8	1	88.89%
Two hands waving	9	6	3	66.67%
Total Result	93	78	15	83.87%

The confusion matrix for this experiment is provided in Table 4. The confusion matrix shows that 15 videos were not recognized correctly out of 93. This fault in recognition is due to the use of only one feature, specifically the contour-based type. This feature captures only the closed boundary (contour) details of the AMI and ignores all other regional (silhouette) details. As shown in Table 4, the confusion of 3 videos occurred in connection with the following actions: bending, jumping in place, and two hands waving. In addition, there was confusion regarding 2 videos for the jumping jack and skip jumping actions, as well as confusion of one video for walking and one for one hand waving actions. Note that this result is the least accurate among all of the experiments we conducted.

TABLE 4. Confusion matrix results of DCDS feature using KNN classifier

Human Actions	Bending	Jumping jack	Jumping	Place jumping	Running	Side jumping	Skip jumping	Walking	One hand waving	Two hands waving
Bending	6		2	1						
Jumping jack		7							1	1
Jumping			9							
Jumping in place	1		1	6			1			
Running					10					
Side jumping						9				
Skip jumping					1		8	1		
Walking					1			9		
One hand waving			1						8	
Two hands waving		2						1		6

3. Only HOG Feature Experiment

In order to find the effectiveness of each feature separately, this experiment was performed using only the HOG feature. This feature is a silhouette-based type and computed by the histogram of intensity gradients of the AMI. The setup parameters were as follows: the number of overlapping windows (NxN) was set to 3x3, and the number of bins (B) was set to 8. For classification, the KNN based on the 1st norm distance was also employed. Note that here the setup parameters used were the same as those employed in the first experiment using the KNN.

As shown in Table 5, the correct recognition rate was 90.32%. This table gives the result for each action in the dataset, as well as the total result for all actions.

TABLE 5. Recognition results of HOG feature using KNN classifier

Human Actions	Recognition Results			
	Videos	Corrects	Wrongs	Correct Rate
Bending	9	9	0	100.00%
Jumping jack	9	8	1	88.89%
Jumping	9	8	1	88.89%
Jumping in place	9	8	1	88.89%
Running	10	9	1	90.00%
Side jumping	9	9	0	100.00%
Skip jumping	10	9	1	90.00%
Walking	10	9	1	90.00%
One hand waving	9	7	2	77.78%
Two hands waving	9	8	1	88.89%
Total Result	93	84	9	90.32%

The confusion matrix for this HOG experiment is provided in Table 6. The confusion matrix shows that 9 videos were not correctly recognized out of 93.

TABLE 6. Confusion matrix results of HOG feature using KNN classifier

Human Actions	Bending	Jumping jack	Jumping	Place jumping	Running	Side jumping	Skip jumping	Walking	One hand waving	Two hands waving
Bending	9									
Jumping jack		8								1
Jumping			8				1			
Jumping in place				8		1				
Running					9	1				
Side jumping						9				
Skip jumping			1				9			
Walking		1						9		
One hand waving	1								7	1
Two hands waving		1								8

These mistakes in recognition were due to the use of only one feature, which was the silhouette-based type. This feature captures only regional (silhouette) details of the AMI and ignores all other closed boundary (contour) details. Although the silhouette implicitly contains the contour (border), it is still not converted into the DCDS feature. Therefore, the results were not better than those in the first experiment using combined features, but they were superior to those in the second experiment using DCDS. This is due to the fact that the contour is implicitly contained the silhouette. The confusion matrix also shows that no more than one wrong recognition for each of the following actions: jumping jack, jumping, jumping in place, running, side jumping, walking, one hand waving, and two hands waving.

C. Support Vector Machine (SVM) Experiments

These three experiments were employed the SVM as a classifier and two features (DCDS and HOG) were also used to recognize the human actions. In the same manner as the KNN experiments, three experiments were conducted using this classifier. In the first experiment, both features were used. In the second, one feature of a contour-based type was utilized. In the third, only the silhouette-based type was used as a feature.

1. Both DCDS and HOG Features Experiment

In this first experiment of the SVM classifier, the two combined features (DCDS and HOG) were employed. The setup parameters for the DCDS in this experiment were as follows: 30 for the FD points and 8 or 22 points as a jump displacement among these points. Also, the setup parameters for the HOG were 3x3 overlapped windows and 8 bins. Both features were combined and used as one feature vector to perform the human action recognition. The SVM based on linear kernel was applied as a classifier.

The total result was achieved a correct recognition rate of 98.88%, as shown in Table 7.

TABLE 7. Recognition results using of combined DCDS and HOG features using SVM classifier

Human Actions	Recognition Results			
	Videos	Corrects	Wrongs	Correct Rate
Bending	9	9	0	100.00%
Jumping jack	9	9	0	100.00%
Jumping	9	8	1	90.00%
Jumping in place	9	9	0	100.00%
Running	10	10	0	100.00%
Side jumping	9	9	0	100.00%
Skip jumping	10	10	0	100.00%
Walking	10	10	0	100.00%
One hand waving	9	9	0	100.00%
Two hands waving	9	9	0	100.00%
Total Result	93	92	1	98.89%

The experimental results for the confusion matrix are shown in Table 8. The matrix shows confusion only between the jumping forward and jumping in place actions. This confusion was due, first, to the similarity in these actions and, second, to the limitations of the linear SVM classifier.

TABLE 8. Confusion matrix of combined DCDS and HOG features using SVM classifier

Human Actions	Bending	Jumping jack	Jumping	Place jumping	Running	Side jumping	Skip jumping	Walking	One hand waving	Two hands waving
Bending	9									
Jumping jack		9								
Jumping			8	1						
Jumping in place				9						
Running					10					
Side jumping						9				
Skip jumping							10			
Walking								10		
One hand waving									9	
Two hands waving										9

Furthermore, in order to make a fair comparison for the effectiveness of these features separately, the same experiment with the same parameters was performed twice using only one of these features (either DCDS or HOG) with the same SVM classifier.

2. Only DCDS Feature Experiment

This experiment was executed to find the effectiveness of the DCDS feature separately (without the HOG). The setup parameters for the DCDS were as follows: the number of FDs points was set to 30, and the jump displacement among these points was set to 8 or 22 points. For classification, the SVM was employed based on the linear kernel. Note that, in this experiment, the setup parameters were the same as those used in the first experiment.

The experiment achieved a correct recognition rate of 85.81%, as shown in the experimental results provided in Table 9. This was due to the use of only one contour-based feature type, which was the DCDS.

TABLE 9. Recognition results using DCDS feature using SVM classifier

Human Actions	Recognition Results			
	Videos	Corrects	Wrongs	Correct Rate
Bending	9	6	3	66.67%
Jumping jack	9	9	0	100.00%
Jumping	9	7	2	77.78%
Jumping in place	9	8	1	88.89%
Running	10	10	0	100.00%
Side jumping	9	8	1	88.89%
Skip jumping	10	9	1	90.00%
Walking	10	9	1	90.00%
One hand waving	9	7	2	77.78%
Two hands waving	9	7	2	77.78%
Total Result	93	80	13	85.81%

TABLE 10. Confusion matrix of DCDS feature using SVM classifier

Human Actions	Bending	Jumping jack	Jumping	Place jumping	Running	Side jumping	Skip jumping	Walking	One hand waving	Two hands waving
Bending	6			3						
Jumping jack		9								
Jumping			7					1	1	
Jumping in place			1	8						
Running					10					
Side jumping				1		8				
Skip jumping					1		9			
Walking						1		9		
One hand waving	2								7	
Two hands waving		1		1						7

Also, the confusion matrix is shown in Table 10. It reveals that 13 videos were incorrectly recognized out of 93. This table shows confusion in recognizing the bending actions occurred for 3 videos. The 3 videos that were confused contained similar actions such as: one hand waving, and two hands waving. Moreover, one video that was confused contained jumping in place, walking, side jumping, and skip jumping actions. These results were slightly better than those for the same feature using the KNN classifier. This was due to the use of different classifier.

3. *Only HOG Feature Experiment*

In order to test the effectiveness of the HOG feature separately, this experiment was performed without using the DCDS feature. The setup parameters were as follows: the number of overlapping windows (*N*x*N*) was set to 3x3, and the number of bins (*B*) was set to 8. For classification, the SVM based on the linear kernel was employed. Note that, the setup parameters used were the same as those employed in the first experiment with the SVM.

As shown in Table 11, a correct recognition rate of 92.22% was achieved. This table provides experimental results for each action in the dataset, as well as the total result for all actions.

TABLE 11. Recognition results of HOG feature using SVM classifier

Human Actions	Recognition Results			
	Videos	Corrects	Wrongs	Correct Rate
Bending	9	8	1	88.89%
Jumping jack	9	8	1	88.89%
Jumping	9	8	1	88.89%
Jumping in place	9	7	2	77.78%
Running	10	10	0	100.00%
Side jumping	9	9	0	100.00%
Skip jumping	10	10	0	100.00%
Walking	10	9	1	90.00%
One hand waving	9	9	0	100.00%
Two hands waving	9	8	1	88.89%
Total Result	93	86	7	92.22%

The confusion matrix is shown in Table 12. It shows that 7 videos were failed to recognize correctly out of 93. These confusions happened twice in the jumping in place action, since of the similarity between the jumping in place and the side jumping. While it happened one time in 5 other actions: bending, jumping jack, jumping, walking, and two hands waving. In the comparison, this result is slightly better that the result of the same experiment that performed using the KNN classifier. This is also due to the behaviour of the classifier itself.

TABLE 12. Confusion matrix of HOG features using SVM classifier

Human Actions	Bending	Jumping jack	Jumping	Place jumping	Running	Side jumping	Skip jumping	Walking	One hand waving	Two hands waving
Bending	8					1				
Jumping jack		8								1
Jumping		1	8							
Jumping in place				7		1				1
Running					10					
Side jumping						9				
Skip jumping							10			
Walking		1						9		
One hand waving									9	
Two hands waving		1								8

V. Conclusions

The research reported in this paper demonstrates optimal human action recognition in terms of recognition rate accuracy by combining two different kinds of features. The first feature concerns the boundary coordinates (contour-based type) and is called DCDS. The second is the regional appearance (silhouette-based type) and is called HOG. Combining these features leads to the formation of a strong complementary feature vector that captures effective discriminant details of human action videos. The KNN experimental results achieved a correct recognition rate of 100%. This result demonstrates that our algorithm promises excellent results in terms of accuracy for human action recognition.

Moreover, the best SVM experimental result achieved was a correct recognition rate of 98.88% of correct recognition rate. This result is very close to the optimal solution and indicates that these combined features can be applied in different classifiers successfully. In addition, the algorithm used in this research applied a new DCDS feature which is very useful for human recognition; it is also low time computation and complexity. It is proven based on the results, in this research, that these features types (contour-based and silhouette-based) are very effective in terms of accuracy, especially when they are combined (e.g., the combination of DCDS and HOG features).

Acknowledgment

The authors would like to thank Dr. Hussain Al-Rizzo, Dr. Marinelle Ringer, and the anonymous reviewers of this paper for their valuable and constructive comments and suggestions. Also, we would like to thank the researchers L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri for the creation of the Weizmann dataset for human action recognition.

REFERENCES

[1] J. K. Aggarwal and M. S. Ryoo, Human activity analysis: a review, ACM computing surveys, vol. 43, 2011, pp. 16:1–16:43.

[2] L. Gorelick, M. Galun, E. Sharon, A. Brandt, and R. Basri, Shape representation and classification using the Poisson equation, IEEE transaction on pattern analysis and machine intelligence, vol. 28, No. 12, 2006, pp. 1991-2005.

[3] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, Actions as space-time shapes, IEEE transaction on pattern analysis and machine intelligence, vol. 29, No. 29, 2007, pp.2247-2253.

[4] T. Whytock, A. Belyaev, and N. Robertson, GEI + HOG for Action Recognition, 4th UK Computer Vision Student Workshop (BMVC 2012 Student Workshop), Surrey, UK 2012.

[5] S. Sadek, A. Al-Hamadi, B. Michaelis, and U. Sayed, Chord-Length Shape Features for Human Activity Recognition, ISRN Machine Vision 2012, 2012.

[6] S. Al-Ali, M. Milanova, A. Manolova, V. Fox, “Chord-Distance Signature Derivatives and Histogram of Oriented Gradient Features for Human Action Recognition,” 16th International Conference on Mathematical Methods, Computational Techniques and Intelligent Systems Proceeding (MAMECTIS '14), Lisbon, Portugal, Aug. 2014, pp. 73-80.

[7] J. Han, and B. Bhanu, Individual recognition using gait energy image, IEEE transactions on pattern analysis and machine intelligence, vol. 28, 2006, pp. 316-322.

[8] E. Mendi, M. Milanova, Y. Zhou, J. Talburt, “Objective video quality assessment for tracking moving objects from video sequences,” Proceedings of the 9th WSEAS international conference on Signal processing, robotics and automation ISPRA'10, Cambridge, UK, Feb. 2010, pp. 121-126.

[9] E. Mendi, M. Milanova, Image Segmentation with Active Contours based on Selective Visual Attention," 8th WSEAS International Conference on Signal Processing (SIP '09), Istanbul, Turkey, June 2009, pp. 79-84.

[10] R. Diaz de Leon, L. E. Sucar, “Human Silhouette Recognition with Fourier Descriptors,” Proceeding 15th International Conference on Pattern Recognition (3), vol. 3, 2003, pp. 709-712.

[11] H. Kauppinen, T. Seppanen and M. Pietikainen, “An Experimental Comparison of Autoregressive and Fourier-Based Descriptors in 2D Shape Classification,” IEEE transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 2, Feb. 1995, pp. 201-207.

[12] R. Gonzalez, R. Woods, and S. Eddins, Digital Image Processing using Matlab, 2nd ed., NJ, USA, Prentice-Hall Inc., Upper Saddle, 2003.

[13] Zhang, and G. Lu, “Review of shape representation and description techniques,” Pattern Recognition, vol. 37, 2004, pp. 1-19.

[14] A. Belyaev, “On implicit image derivatives and their applications,” Proceedings of the British Machine Vision Conference, pp. 1–12, 2011.

[15] N. Dalal and B. Triggs, “Histograms of Oriented Gradients for Human Detection,” Proceeding of IEEE Conference in Computer Vision and Pattern Recognition, 2005, pp. 886-893.

[16] M. M. Deza and E. Deza, Encyclopedia of Distances, 2nd ed., Berlin: Springer, 2009.

[17] R. O. Duda, P. E. Hart and D. G. Stork, Pattern Classification, 2nd ed., New York: Wiley-Interscience, 2000.

[18] A. Ben-Hur and J. Weston, A User’s Guide to Support Vector Machines, vol. 609, Clifton, NJ, USA, 2010, pp. 223-239.

[19] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," Data Mining and Knowledge Discovery 2, 1998, pp. 121-167.

[20] C.-C. Chang and C.-J. Lin, "LIBSVM: A Library for Support Vector Machines," ACM Transactions on Intelligent Systems and Technology, vol. 2, no. 3, 2011, pp. 27:1-27:27. Software available at:

[21] http://www.csie.ntu.edu.tw/~cjlin/libsvm

Salim Al-Ali is a PhD graduate student in the Integrated Computing program of the Computer Science Department in the University of Arkansas at Little Rock (UALR). He received the first Master of Science degree from Computer Science Department, Baghdad University, Iraq in 1995. Also, he received the second Master degree in Applied Science from Applied Science Department in the UALR in 2014. He is working as a Teacher in Duhok Technical Institute at Duhok Polytechnic University in the Kurdistan Region. His research fields are in Computer Vision in the fields: Human Action Recognition, Image and Video understanding, Data Mining, Classification and Clustering.

Mariofonna Milanova has been a Professor of Computer Science Department at the University of Arkansas at Little Rock since 2001. She received her MSc in Expert Systems and AI in 1991 and PhD in Computer Science in 1995 from the Technical University, Sofia, Bulgaria. She did her post-doctoral research in visual perception at the University of Paderborn, Germany. She has extensive academic experience at various academic and research organizations in different countries. She serves as a book editor of two books and associate editor of several international journals. Her main research interests are in the areas of artificial intelligence, biomedical signal processing and computational neuroscience, computer vision and communications, machine learning, and privacy and security based on biometric research. She has published and co-authored more than 70 publications, over 43 journal papers, seven book chapters, numerous conference papers and two patents.

Agata Manolova is an Assistant Professor at the Faculty of Telecommunications at the Technical University of Sofia, Bulgaria. She obtained a Ph.D. in Computer Science in 2011 from the University of Grenoble, France. Dr. Manolova conducted post-doctoral research as a Visiting Scholar on a Fulbright Grant in the University of Arkansas at Little Rock, USA. Her domains of interest are Pattern Recognition, Computer Vision, Virtual Reality, Image and Video processing. Dr. Manolova has participated in several scientific projects both national and international concerning object recognition in video sequences, developing of a smart wall for disabled people, compression of multispectral and hyperspectral images. She is teaching subjects such as Multimedia systems, Video and Audio engineering, Computer vision and Machine Learning, Signal and Image processing.

Victoria Fox is an Assistant Professor at the University of Arkansas at Monticello, United States and is a recent graduate of the University of Arkansas at Little Rock with a doctorate in Applied Science. She is currently researching computational methods in solving partial differential equations, image processing in remote sensing, and the integration of natural resources management with geospatial technologies.